Spring 2019

Microscopia: A Virtual Museum of Organisms Created with SEM Photogrammetry

Ian Donnelly

MICROSCOPIA: A VIRTUAL MUSEUM OF ORGANISMS

CREATED WITH SEM PHOTOGRAMMETRY

By

Ian Donnelly

A Thesis Submitted in Partial Fulfillment

of the Requirements for a Degree with Honors

(New Media)

The Honors College

University of Maine

May 2019

Advisory Committee:

Dr. Sofian Audry, Professor of New Media, Advisor
Dr. David Gross, Professor of Honors
Dr. Valerio Rizzo, Professor of Neuroscience (University of Palermo)
Dr. Joline Blais, Professor of New Media
Aaron Boothroyd, Professor of New Media

ABSTRACT

For many, insects and other minute organisms are regarded with disgust, disinterest, and labeled as “pests.” Often they are either in the way or out of sight. There is, however, more than meets the eye. These organisms not only play a monumental role in the world’s ecosystems—a truth that has become ever more apparent in the face of global climate change—but they also have astounding physiological complexities that humans can’t quite see without the aid of a microscope. The scanning electron microscope (SEM) offers a highly detailed view, the ability to see specimens from all angles, and a glimpse into the intricacies of creatures that walk amongst us every day. Unfortunately, this technology is expensive to use and is rarely experienced by the public.

Microscopia is an attempt to emulate the eye-opening experience of seeing organisms through an SEM and take it one step further—allowing users to seemingly shrink down to their size. By capturing SEM images of specimens in various rotational series and processing them through photogrammetry software, 3D models are generated. The models are displayed in two contexts, both built with Unity game engine. The first is a virtual reality experience that incorporates narration, sound, and a simulated shrinking of the user. The second is an augmented reality mobile companion app that allows users to place the models in their world and effectively bring the experience home with them.

ACKNOWLEDGEMENTS

This project could not have been made possible without the open arms of individuals and organizations at the University of Maine and beyond.

First and foremost, I would like to thank my thesis committee for offering both their knowledge and time to provide critique, support, and refined purpose relating to this endeavor. Each member played an integral part in shaping the final product.

My advisor, Dr. Sofian Audry, provided consistently insightful feedback and offered a unique perspective as an artist and mentor. He allowed me to audit his capstone course to get additional feedback from other New Media seniors, which led to many of the ideas that make Microscopia successful.

I would like to thank Kelly Edwards, Electron Microscopy Lab Manager in Murray Hall, for training me in the use of the SEM. He showed patience despite my countless blunders and enthusiasm towards an idea that I, at first, had no proof I could bring to fruition.

Dr. Allie Gardiner, Professor of Arthropod Vector Biology, deserves great thanks for allowing me access and use of specimens from the entomology closet in Deering Hall. Additionally, her love of and fascination with insects opened my eyes to the possible educational contexts of this project.

I would also like to thank Dr. Emma Perry, Professor of Biology at Unity College, for finding, mounting, and educating me about several tardigrade specimens. Without her help, I would not have been able to show just how effective SEM photogrammetry can be.

Dr. Valerio Rizzo deserves particular recognition and gratitude for supporting this project for nearly two years. Once a stranger in an online photogrammetry software forum, he has since become a mentor, a source of inspiration, and a friend. He also assisted greatly in the reconstruction and modeling of the fire ant model.

Photogrammetry software company, Capturing Reality, gave me free use of their industry-standard software, Reality Capture, for the duration of this project, given the rights to feature it in their marketing campaigns. I have been grateful for their constant support and excitement every step of the way, as well as the support and excitement of their growing online community.

Arturo Camacho provided the equipment, skills, and resources to scan the IMRC’s AP/PE Space. In this way, he helped me add a feature to this project that I thought was unattainable and I am incredibly grateful for his commitment.

I would like to thank both Aaron Boothroyd and Dr. Joline Blais for creating the adjacent Focus Ring and Stillwater labs and for granting me 24/7 access to the VR equipment shared between them. This was especially appreciated in times of desperation during development, in which the 24/7 access was truly utilized.

Emily Gagne, a graduating senior majoring in Biology, took time out of her busy schedule to offer her voice as the narrator of the Microscopia VR experience. I thank her for her time and dedication to capturing the tone desired.

I deeply thank The Center for Undergraduate Research for providing both the funding that made Microscopia possible and a platform, in the form the 2019 Student Symposium, to reach a wider audience of students, professors, and the public.

I would like to express my gratitude for Dr. David Gross, as well, for instilling in me many of the ideas that brought me to this one. His unique interpretations of David Foster Wallace's work and William Blake's poetry, among other works explored in Honors 212, evoked within me a closer inspection of the world's intricacies and wonders.

Lastly, I would like to thank two groups that indirectly contributed to the VR experience. The first is Sigur Rós, an Icelandic ambient/post-rock group, who allowed me to use their track "tónandi liminal" as the soundtrack. The second is ScanSource 3D, a Sketchfab creator, who created a Creative Commons 3D model of a penny that I used to provide scale.

TABLE OF CONTENTS

LIST OF FIGURES

INTRODUCTION

This Honors thesis details the artistic purpose, research, and methodology involved in creating Microscopia, a virtual museum in VR and AR consisting of organisms reconstructed with SEM photogrammetry. Three primary sections in the body of this thesis pertain to different aspects of the process. The Artist's Statement discusses the background interest, initial ideas, and narratives that culminated in the goals of this thesis. In the Literature Search section, texts that supplemented the methodology discussed are summarized in the context of the project. The Methodology section outlines each step and decision involved in the project. Being an interdisciplinary pursuit, this thesis involves both technical methods (such as microscope use and programming) and visual design. Appendices are included with samples of design elements, screenshots, renders, code, and the script of VR narration.

ARTIST'S STATEMENT

Nearly two years ago, I sat in my cubicle in a Portland non-profit as a marketing intern, pondering what my Honors thesis might be. After a plethora of ideas drifted through my mind, I thought about how enamored I had been with images captured with an SEM that I had seen on the internet. Blood cells, tardigrades, viruses – all rendered in otherworldly detail. From a young age, I have been interested in the unseen processes and creatures that influence our living world. I remember, in an early memory, picking up a handful of dirt and wondering where it came from and what might live amongst its tiny particles. At the University of Maine, I have had the opportunity to enrich my knowledge of the natural world with a minor study in biology. Still, the older I get, the more I feel myself losing that unashamed, childlike curiosity of palming through dirt to busyness and distraction. When I first began thinking about an SEM-related thesis project, I thought that perhaps this would bring some of that curiosity back again.

Wanting to be a graphic designer at the time, I decided that I would take pictures of natural substances with an SEM and incorporate their various textures in digital art pieces. Over the following weeks, however, I began exploring other avenues. I did not even truly know what photogrammetry was when I considered that there might be a way to create 3D models from SEM imagery, but I decided to follow that idea to its conclusion.

This led me to a forum thread for photogrammetry software Autodesk Remake, in which someone proposed the idea but abandoned it, despite encouragement from Dr.

Valerio Rizzo. Desperately wondering how I might go about starting this project, I managed to contact Dr. Rizzo via his workplace email address and get the answers I needed to get my feet off the ground.

My goal then was to reconstruct a specimen in 3D. Each additional component of the project, including the augmented and virtual reality, the user interfaces, and the C# scripting, came together as I developed as a creative over the past two years. As the project developed, so did its purpose. It began as an exercise in attempting to push the limits of photogrammetry, but soon evolved into a way of coming face to face with organisms that we disregard in our day to day existence. More than anything, I want this creative thesis to give users the harrowing sense that they are, for the first time, on equal terms with these miniscule yet infinitely complex beings and learn about them in an entirely new educational context.

LITERATURE SEARCH

The articles, papers, and tutorials detailed here represent the core sources of knowledge required to complete this project. These will be properly cited in the References section. It is worth noting that there were countless instances, especially while scripting, that forums, FAQs, and documentation pages were consulted to solve problems related to C#, Vuforia SDK, and SteamVR in Unity game engine. As many of these sources were innumerable, ephemeral, and pertinent to specific aspects of this project alone, they will not be listed in this thesis.

Full Photogrammetry Guide for 3D Artists

This guide was written by Vlad Kuzmin, a photogrammetry expert and Reality Capture enthusiast. It details each step from planning out a photogrammetry scan to publishing the finished model online. Although I did not replicate the complete workflow, partially due to time and funding constraints that prevented me from investing in all the necessary software, I learned much and shaped the workflow to my needs. The most helpful aspect of the guide, for me, was the documentation of the less apparent features and settings of Reality Capture. Given both the breadth and specificity of this guide, it is easy to feel like you can photogrammetrically scan just about anything.

Unity Photogrammetry Workflow

Whereas Vlad Kuzmin's guide covers photogrammetry for the general purpose of sharing models with others, this guide written by Sébastien Lachambre, Sébastien Lagarde, and Cyril Jover—all researchers at Unity—focuses on the production of game-ready models. It covers more or less of the same material at first but contains specific sections about preparing the models for a virtual environment in Unity engine. Reality Capture is also the photogrammetry software of choice in this guide and it contains all the necessary details about exporting complete meshes. It also covers simplifying, texture baking, and UV editing. The general goal in this workflow is to create a reconstruction with high detail and simplify it while retaining as much of that detail as possible.

SEM-Microphotogrammetry, A New Take on an Old Method for Generating High Resolution 3D Models from SEM Images

Before I had completed any one of my models, the only instances of detailed models created with SEM photogrammetry I could find online were those published by the Natural History Museum on the 3D model hosting site Sketchfab. The heads of three different insects can be viewed and downloaded on the site. It was not until I had established my own workflow that I came across this article, whose primary author is Dr. Alexander D. Ball, a researcher and electron microscopy expert at the Natural History Museum. I was surprised to find that much of my own SEM procedure mirrored the one in this article, albeit with less pictures captured to account for the cost of scan time. Beyond that, though, Dr. Ball and his co-authors contextualize the use of SEM photogrammetry in ways I had not considered and supplement their methods with extensive research and background knowledge. In many ways, this article filled in the gaps in my own knowledge and paved the way for me to experiment even further in this uncharted territory.

HTC Vive Tutorial for Unity

Even though "VR" has been a buzzword for years now, there is a surprising lack of documentation regarding VR development in Unity game engine. This is mainly due to there being two versions, old and new, of the SteamVR developer kit. The new version is more user-friendly but has some unresolved issues and even less documentation than the old version. For this reason, the Unity VR development community is split in two. Fortunately, Eric Van de Kerckhove at Raywenderlich has written two comprehensive tutorials, for both versions of SteamVR, that take beginners through the complete setup of a VR environment. The tutorials contain sample projects that require the developer to add interactivity via C# scripting. By the end of these tasks, VR development in Unity becomes less of a daunting beast and more of a playground.

Introduction to Ground Plane in Unity

Out of all the technologies used in this project, the most familiar was the Vuforia augmented reality software development kit for Unity. I had previously created Vuforia projects such as an AR business card and a heart rate visualization, both using image target recognition. One feature of Vuforia I had not gotten the chance to implement was Ground Plane, which allows users to place models on any surface in the camera view. Much like the other tutorials mentioned in this literature search, this introductory guide walks through each of the necessary steps to create a basic application with this technology. Each facet of the Vuforia SDK is well-documented in the Developer Library, making AR development accessible to just about anyone with a working knowledge of Unity engine.

METHODOLOGY

This section will recount the methods of each phase of this thesis project. The phases regarding the creation of the models had consequential steps, while those regarding development and design often involved constant evaluation of which tasks and features were to be undertaken next.

Specimen Preparation

Acquisition – Although only three specimens are shown in the final VR and AR products of this thesis, many other specimens were acquired as well. A European fire ant, the first specimen used in this project, was left over from a previous EM lab session and available for use. The next specimen, a black vine weevil, was taken from a collection of insects kept in Deering Hall, acquired by various entomology classes over the years. These insects are either mounted on a pin through their abdomens or adhered to wood glue on the end of a paper stub. Some of the specimens date back to as early as 1921, indicated by tags beneath them.

Given that specimens mounted on a pin would be noticeably damaged at higher magnification, wood-glued specimens were preferable. To remove the glue, specimens were carefully transferred into a petri dish of isopropanol by clasping the attached paper stubs with fine-tipped steel forceps. A wooden toothpick was used to prompt separation from the paper. Many times, too much force would cause the dry specimens to break in various ways. The black vine weevil, for example, lost its two forelegs in the process. Ideally, specimens should be caught live such that they will be free from any previous mounting and will not be as brittle.

For the tardigrade specimen, Dr. Emma Perry of Unity College offered her assistance. She had gathered tardigrades from lichen growing on a wood pile near her home, where she often finds specimens of the genus *Echiniscus*, and brought them into the lab in a small, plastic capsule.

<u>Mounting</u> – Specimens were mounted on standard 12mm aluminum stubs with adhesive carbon tape disks. Posing the specimens on the tape proved to be one of the most difficult tasks of this phase. The only tool that could be used to lift and position the specimens without damaging them was an eyelash brush, consisting of a single swine eyelash glued to a wooden toothpick. With the petri dish of isopropanol and unmounted specimens under the view of a stereo microscope at low magnification, the tip of the eyelash brush was threaded through the legs of the specimens. At this point, they were guided with a steady hand over to the aluminum stub and positioned vertically from the center. In most cases, this resulted in dropped, damaged, and wrongly-positioned specimens. Successful mounts, however, did occur. In the case of an unsteady mount, silver-nitrate paint was brushed around the base of the specimen for added stability.

Before mounting, Dr. Perry used a critical point dryer to preserve surface structure that would otherwise be damaged by changing from liquid to gaseous state in a vacuum. Once dried, the capsule of tardigrades was opened under the stereo microscope at an even higher magnification. Tardigrades are barely visible to the naked eye, so mounting them is an arduous task. An eyelash brush was used in this case as well, but in a different manner. Dr. Perry was able to lift the specimens using only the surface tension between the eyelash and the tardigrade.

Coating – The conductivity of a specimen heavily dictates the resulting SEM image. Specimens that are not as conductive often appear darker and tend to exhibit what is called “charging”. This is when electrons build up on a surface and produce a significantly brighter area in the SEM. The effects of charging can be seen on the fire ant model, where the tips of its antennae look brighter than the rest of its surface. To reduce this effect, specimens were made more conductive by covering them in an approximately 10nm thick layer of gold-palladium using a sputter coater in the EM lab. The sputter coater works by creating a vacuum and then pumping in argon gas, which flushes out moisture in the air and acts as a transport medium for the gold-palladium.

Photogrammetry

Microscope Use – Once a specimen was prepared, the SEM chamber could be vented and opened after pushing the release valve. The stage was then loosened with an Allen wrench, allowing the stub to be inserted, and then tightened again. Once the chamber was closed and gently pushed flush against the exterior, the chamber was evacuated and returned to a vacuum. Afterwards, the release valve on the chamber was pushed back in, the filament was turned on, and the beam was set to 10kv. With these steps completed, the chamber came into view on the monitor.

This SEM, the AMRay 1820, has various knobs that can translate and rotate the stage on each access. Almost always, the stage needed to first be adjusted to set the stage to an initial side-on view of the specimen. Each specimen required different levels of magnification, but in general the specimens were magnified to roughly fit the frame. The magnifications of the featured specimens were roughly 35x for the fire ant, 30x for the weevil, and 300x for the tardigrade. These values are not exact as photogrammetry does not require the same magnification across a photo series.

A practice that was not honed until nearly the end of the project was focusing the images. The basic means of focusing an SEM image is using the focus knob at the current magnification. To get an even finer focus, clicking the "partial field" button on the control pad allows the viewer to see an even closer view in an added window on the monitor. Increasing magnification in partial field mode magnifies the image within the window while still displaying the entire specimen at normal magnification outside of it.

This often produces crisp images, but there is still one factor that could dampen the quality of the image: beam astigmatism. The SEM has rays that propagate in two perpendicular planes and sometimes these rays have different foci. To align them, there is a component called a stigmator that imposes a weak magnetic field on the rays. It can be controlled with a dual knob on the control panel; each part controlling one of two planes. The knobs are manipulated in tandem until the image looks to be in optimal focus. In this way, calibrating the stigmator is not much different than using the focus knob.

Capturing Images for Photogrammetry – Following the advice of Dr. Valerio Rizzo and the photogrammetry guide by Vlad Kuzmin, I devised a workflow for capturing SEM images for the purpose of photogrammetry. Starting with a side-on view of the specimen and a stage rotation of 90 degrees, images were captured along a radial arc every 9 degrees. Once these forty images were taken, the stage was angled at 45 degrees in order to expose the top of the specimen. Another radial arc was made, this time capturing images every 18 degrees. About every fifteen images, the files were transferred from the EM lab's PC to my own laptop for alignment in Reality Capture. This provided an estimate of what features needed more images captured of them, retroactively changing the workflow. After additional images were captured to fill in missing data, the total number of images captured usually reached about 70 to 80.

Ideally, more arcs would be made around each specimen to increase image overlap. At least 60% overlap from one image to the next is a good rule of thumb in photogrammetry image capture. In a standard photogrammetry use case, where the individual is taking pictures around the subject with a DSLR, it does not take long to capture hundreds of photographs. In the SEM, however, each image can take up to two to three minutes to capture—even longer at some settings. Given the lab's rate of more than $50 per hour of SEM usage, a compromise had to be made between the number of images captured and the quality of those images.

The image output settings used throughout the project were 2048x2048 resolution, a point average (smoothing) setting of 4, and TIFF file format. Images were captured in less than two minutes on average and contained little noise. In some cases, images were batch white-balanced in Lightroom before being imported into Reality Capture.

<u>Reality Capture</u> – As aforementioned, Reality Capture was used throughout the image capturing process to continually align the images. The software is a proverbial

“black box” at this point in the process; taking the 2D inputs and converting them into a 3D point cloud output using computer vision and various algorithms, without any sort of manipulation on my end. If enough features are identified on an image that match up with the same features on others, it is marked as aligned and is used in the computation of the 3D model. If not, however, the image will be left unaligned. This is where manual manipulation was required on my part.

Reality Capture contains “control points,” which allow the user to point out features in each of the images and force the software to re-examine them in the context of the series. Control points can be used not only on unaligned images, but misaligned images as well. When misaligned images are marked with control points, warning icons are often displayed next to the point to indicate differences in position. Once “Align Images” is pressed again, the images are forced into a new alignment and, most of the time, these warnings disappear. To map out an accurate surface topography of a specimen, it took about twenty control points on average.

On rare occasions, aligned images generated inaccurate features in the resulting mesh. For example, three images of the black vine weevil produced a smaller head attached to the side of the model’s proboscis. Hours of control point manipulation failed to remedy this issue. Per Dr. Rizzo’s suggestion, the point lasso was used to select points within the aberration and by selecting the “Find Images,” the images causing the problem were identified. By marking these as “disabled in mesh,” the issue was resolved.

<u>Mesh Export</u> – Once an image set seemed to be in the best alignment possible, “High Detail” reconstruction was selected, starting the calculation of the 3D mesh. This took roughly half an hour with a Dell XPS 15” laptop containing 16 GB of RAM, an Intel

i7 processor, and a Nvidia GeForce GTX 1050 graphics card. The resulting mesh was white after reconstruction, as it was untextured. Clicking “Texture” triggered the texturing process, which took another fifteen minutes or so.

High detail reconstruction results in a mesh with a high polygon count that is not optimized for real-time rendering in Unity, especially on hardware like an older phone. Both photogrammetry guides in my literature search detail the processes of simplification, retopology, and texture baking in the creation of an optimized mesh. Without any experience with the different software required and a tight budget of both funding and time, a simpler route was taken. By using the “Simplify Tool” in Reality Capture, meshes were reduced from more than 3 million polygons to 60k polygons. This reduces the number of faces on the mesh without sacrificing much quality. After simplification is complete, texturing the mesh again maps the high detail texture to the lower detail model. Although this quick procedure is likely more destructive to the model’s quality than those described in the guides, it produces models with negligible differences from the high detail reconstruction while taking a much shorter amount of time.

Cleaning Up the Models – Each textured model exported from Reality Capture was imported into the 3D modeling software Blender for final touch-ups. In the cases of the black vine weevil and the tardigrade, pieces of the stub beneath the specimen appeared in the final mesh. The vertices of these extraneous pieces were simply selected using the “Lasso select” tool and deleted.

There were also visible ridges created by slight misalignments in the meshes. By using Blender’s sculpting tools with the smoothing setting turned on, these ridges were easily smoothed out. The only issue is that many of the ridges were an untextured gray color that, when smoothed out, were even more apparent on the finished model. To fix this, the "Clone Brush" tool was used to copy the texture adjacent to ridge and blend it across the untextured area.

The European fire ant model received some special treatment from Dr. Valerio Rizzo early in the project, in which he smoothed out some of the jagged features of the model, remodeled some of the setae (small hairs) that were lost in reconstruction, and made other general aesthetic improvements. The model created without his expertise was not nearly as true to the actual specimen. It is possible the specimen would not have been included in the final product in that state.

<u>Technical Challenges</u> – As mentioned previously, only three specimens were included in the final product. Six specimens were attempted, but a multitude of issues prevented the other three from reaching the level of detail and mesh quality desired.

One of the specimens, a beetle, failed due to inconsistencies in conductivity between sessions. As only half of the images could be captured on one day and the other half on another, there were two resulting sets of images that were photogrammetrically incompatible. The first exhibited almost no charging effects, while the second had charging on the ends of each of the beetle's legs. Another layer of sputter coating failed to reduce the charging and the specimen was subsequently abandoned.

A different species of tardigrade was attempted before the *Echiniscus virginicus* specimen that ended up in the final product. It was much flatter than the *Echinicus* specimen, producing two sides that aligned decently well but would not align with each other. Despite the use of many control points to try and merge the disparate components, they simply would not comply. Hypothesizing that a rounder specimen might yield a better result, I made a request to Dr. Emma Perry. This hypothesis proved to be true in the reconstruction of the *Echiniscus* tardigrade.

VR Development

The Microscopia virtual reality experience was developed in Unity engine using C# scripting and SteamVR. The goal of this experience is to allow users to feel like they have shrunk down to the size of the organisms and get a chance to learn more about them through visual and audial perception. With the concept of Microscopia as a whole and the methods behind it being so alien to the everyday person, I felt that this experience should be simple in respect to both environment and interaction.

Environment – The primary environment in this experience is that which exhibits the models. Initially, the space was going to be modeled like a modern museum. As time went on, however, I realized that it would be difficult for me to make this look convincing, let alone photorealistic, which would distract from the hyper-detailed models that are intended to be the focal point. So, I experimented with the exact opposite of having these models displayed in a room. Stepping into VR with the organisms in an empty environment, I realized that it felt almost as if they existed on this ethereal plane.

Doubling-down on this concept, I made the environment completely black by changing the ambient scene lighting to zero and equipped each model with a soft glow from beneath them. The models themselves were given an unlit texture, meaning that their texture is not affected by light. This gives them the same appearance as displayed in Reality Capture.

Whereas the traditional museum exhibit concept would give the user the sense that there is a world outside the walls, this dark space is intended to feel vast, infinite, and empty. That emptiness, however, provides a stark contrast that makes the models all that much more striking and colossal. In our daily lives, we have so much else to look at and distract ourselves with. Here, the user is forced to confront what they would normally deem insignificant.

The specimens are labeled in this space with both their common and scientific names. With no walls or structures, these labels are suspended next to the models. They are rendered in white text with the font Barlow Semi-Condensed; the font used throughout the project. The use of this font and the project's color palette will be discussed more in the Design section.

Both the locations and sizes of the specimens changed several times over the course of development. At first, the specimens were about player height. The microscopic features in this view, though, were too small to easily make out. Then, I tested making them absolutely gigantic—about 30ft tall in virtual space. The problem with that scale was that the user would mostly be seeing the bottom part of the specimens with little view of their heads. Additionally, the textures looked a little pixelated and this was distracting. Scaling the models down to about 12ft tall proved to be the right fit. The models were large enough to see their defined details in crisp textures and their upper features were easily visible without having to look straight up. Additionally, having the models be larger than the user imbues them with a sort of power and dominion in this space.

I chose to make all of the models the same scale so that they would have the same visual weight. Of course, this is misleading as the specimens are different sizes in real life; the black vine weevil is 5mm in length while the tardigrade is 0.3mm in length. To visualize the true scale of these specimens, a table—in the form of a glowing rectangular prism—was added into the space with the models displayed in real-life proportion to each other. Although the sizes of the specimens in millimeters are shown above them, it would still be difficult to imagine how small they are in the scale of our world. For this reason, I added a Creative Commons licensed 3D model of a penny from ScanSource3D into the space and scaled it in proportion to the specimens.

<u>Transition Space</u> – About halfway through development, I wondered if it might be possible to scan the space in which the VR experience is physically taking place. This would provide a means of transition between the real world into the microscopic world and a simulated sense of actually shrinking.

The project would be presented at New Media Night at the end of the year in the IMRC's AP/PE Space on campus. This exhibition space is dark, with black walls and a white, nearly featureless floor; not exactly ideal for a process that requires the recognition of features. Nonetheless, I allotted a short amount of time to attempt the feat. Initial tests in low lighting shot with my own DSLR proved to be failures due primarily to noise in the images. I quickly realized that if this scan was to be successful, I would need to sufficiently light up the space to a point where the camera's ISO could be set as low as possible.

With the help of Arturo Camacho, an Intermedia MFA student at the University of Maine, I was able to do just that. Arturo created a camera setup of four strobes synced to his own Nikon DSLR with a 35mm lens. To capture the space, the setup was moved along the perimeter of the room at intervals of about ten feet. At each interval, four pictures were taken perpendicular to the closest wall at varying degrees of tilt. The tilt angles were not exact, but they roughly intended to capture the floor, the floor and part of the room, part of the room and the ceiling, and then solely the ceiling. This process created more than 200 images in total that were batch white-balanced in Lightroom then aligned in Reality Capture. Additional photographs were taken of features that needed more details and added to the alignment. Control points proved to be necessary once again, as the images aligned in several different parts at first. Similarly to the specimens, the AP/PE Space mesh was reconstructed in high detail, simplified, and then textured before export.

The resulting mesh was acceptable, especially with a darkened texture. My only concern was that the floor was full of holes and ridges that resulted in a lack of real-life detail. The simplest solution to this problem was removing the floor by filtering it out in Reality Capture and placing a flat, textured plane as the floor in Unity instead. A pale, concrete texture replicated the real floor convincingly enough that further scanning of the space was not necessary.

Teleportation – One of the most vital aspects of VR interaction is how the user navigates around the environment. Knowing that I would not have the entire room to myself at New Media Night and possibly other events, I wanted the user to feel like they were navigating around a huge space while restricted to a radius of only a few feet in the real world. In looking at how many VR experiences accomplish this, it was clear that teleportation was the simplest mechanism. Users hold down a button on a remote that displays an arced path to a possible future position and then they release the button to go there. Fortunately for me, this functionality was already built into the open-source SteamVR tutorial. I adjusted the aesthetic aspects of teleportation to my liking and simply copied it into my own project. I then created a teleportation area in the Unity scene and set its size as the imposed boundary of the exhibit. Zones where teleportation is disabled were placed within the specimen models to prevent users from accidentally ending up inside them. These zones are indicated by a red teleportation arc.

Gaze Selection – Gaze selection was an interaction mechanism that had to be scripted from the ground up. This mechanism was preferable to users having to press buttons or perform gestures as turning one's head to look at a point of interest is already an innate action. Gaze selection was created primarily for triggering audio narration. Essentially, users look at a ring labeled with a certain anatomical feature of a given specimen. The line of sight is indicated by the reticle, a small white dot, always positioned in the center of the user's view. When the ring is in the line of sight, it begins to radially fill up with color to indicate selection. If the user looks away, the ring's color begins to deplete. After two seconds in direct gaze, the ring is filled completely, a "play" icon appears, and an audio narration about that anatomical feature begins playback. The color then begins to dissipate over the length of the narration.

To create the gaze selection mechanism, a ray was added to the center of the user's head extending directly outwards. A ray is an invisible line in 3D space that is able to generate feedback on what it collides with. In this case, the ray was scripted to return a number corresponding to the designated layer the collided object is in. Each selectable ring, of which there are nine, was set to its own layer with a unique label. When the ring in the layer set as “UIRingSelect1” is hit by the ray, it changes the global variable “lookingAt” to one. This variable is accessed by a script on the ring itself, which gradually increases the color fill amount of the ring when “lookingAt” is equal to one at any given time. When the value is not equal to one, it gradually decreases.

With nine rings each containing their own narration, it was necessary to make sure that when one is triggered, all of the others turn off. The use case in this scenario is when the user decides halfway through one narration that they would like to listen to another. If the first narration kept playing after the selection of the second, this certainly would not be ideal. Consequently, each respective ring’s script contains an “isPlaying” boolean and references the same variable of all of the other rings. If the value of that boolean changes to true in one ring while it is already true another, then value is set to false in the ring that was previously playing. The boolean “isPlaying” directly controls the playback of the ring’s related audio file.

<u>Shrinking</u> – The microscopic exhibit is represented in the virtual AP/PE Space as a black cube at the user’s feet. A real-world analogue was created as well to further blur the line between realities. In order to trigger the transition from the first “level” to the next, the user must use gaze selection on the cube. This action fills up a ring around the cube that, once completely filled, catalyzes the shrinking process.

The player gameobject’s scale value gradually decreases and the opacity of a black plane in front of the user’s vision increases, fading the view to black. The blackness ensures a smooth transition and eliminates the chance for the user to see the degradation of floor texture quality upon shrinking. When the view is completely black, all of the gameobjects in the first “level” are disabled and those in the second (such as the specimen models, labels, etc.) are enabled. The black plane then decreases in opacity again, revealing the microscopic exhibit.

Sound – From the moment I first stood next to a specimen in VR, I had a sense of what kind of sounds should inhabit the space. I set out to find ambient works that would supplement the experience, not distract from it. For me, this meant no "voices," human or instrumental. I also wanted the tone to be uplifting, meditative, and almost holy. Fortunately, one of my favorite music groups, Sigur Rós, has produced tracks with those very qualities. Their recent album, "tónandi liminal," is an hour-long ambient remix of their previous works. As I listened through it, I found pieces that I felt would fit perfectly in the Microscopia VR experience. These were cut from the hour-long track and looped in Adobe Audition, creating two separate tracks for the two virtual spaces. While the player is in the virtual AP/PE Space, they hear a calming track with subtle vocal elements. When they begin shrinking, however, this track fades out and a more bright and choral track takes its place. This new track repeats throughout the rest of the experience but decreases to a lower volume after the shrinking sequence ends as to not overpower the narration.

Narration was recorded with a Shure SM57 condenser microphone into Adobe Audition and edited for gain and duration. Emily Gagne, a senior Biology major at the University of Maine, provided her voice for the narration. Information about the specimens was found in multiple online resources that are listed in the References section. A written script of the narration can be found in Appendix C.

AR Development

An augmented reality mobile application was developed as a means for users to interact with the museum after they have tried the VR experience or if they did not get a chance to try it. Whereas models in the VR context appear in a virtual exhibition space, models in the AR context are projected into the real world as seen through a camera view. Two different AR viewing modes were developed for the application using the Vuforia software development kit for Unity.

Card Mode – This viewing mode takes advantage of Vuforia's image target detection. Three-inch by five-inch specimen cards were designed in Adobe Illustrator and printed on matte cardstock. High quality .jpg images of these cards, exported from Illustrator, were uploaded to my Vuforia Developer database. Then, the database was downloaded and imported into Unity. Each card was added into the Card Mode scene as an Image Target gameobject. The AR Camera gameobject searches for the Image Targets when the scene is playing. When one enters the view, the camera displays whatever gameobject is set as a child of it in the Unity Editor. By simply placing each specimen model as a child object of its corresponding Image Target, a basic AR experience can be created.

Surface Mode – Microscopia cards will be distributed at exhibition events and made available online, but what if users simply do not have them at the time? It was vital that this application work with and without additional materials. Vuforia's Ground Plane detection offers a suitable solution to challenge. AR Camera, Plane Finder, and Ground Plane Stage gameobjects were added to the Surface Mode scene. Then, the specimen models were added as children of the Ground Plane Stage. When the scene was played, an icon indicating plane placement appeared in the center of the camera view. Upon tapping the screen, all of the models were placed on the corresponding surface.

A script was created, along with some UI buttons designed in Illustrator, to select which specimen is displayed in the camera view. By default, the European fire ant is the specimen placed when the screen is tapped. When a different specimen's button is pressed, however, the script disables the European fire ant gameobject and enables the gameobject of the corresponding specimen. The specimen selection buttons are displayed as white circular outlines containing white silhouettes of the specimens. To indicate which specimen is selected, the selected specimen's button is at 100% opacity, while the other specimens' buttons are at 60% opacity. Additionally, the common name of the selected specimen is displayed at the top of the screen.

<u>User Interface</u> – The application consists of a title screen, a menu screen, an information screen, the two viewing modes, and an instructional loading screen for each. The layout for each screen is minimalistic and indicative of the simple user pathways through the application. There are not many features in the application, so it was best to give as much emphasis to the ones that are there.

For the menu screens, the UI Canvas gameobject was used to position icons and text in a scalable manner. This method does not work in scenes with an AR Camera, however, so the "OnGUI" scripting method must be used instead. OnGUI is a means of user interface creation through script alone. The biggest flaw of this method is that the resulting interfaces are not inherently scalable. To make them compatible with any screen size, the dimensions of each UI element must include a calculation based on screen size.

This way, the elements grow and shrink in proportion to the device used, scaling across all phones and tablets.

It should also be noted that the Microscopia app contains animated UI elements. Using Unity's animation tab, screen transitions were created to fade elements in and out. Animation was also used to move the viewing mode icons along the y-axis when fading them in to make the transition more dynamic and draw attention to the focal point of the screen.

Design

The overall design aesthetic of Microscopia as a product developed slowly over the creation of its various forms. It is easier to show what design choices were made than explain them—which is why images and graphics from the project are displayed in Appendix A—but, in general, I strived to abide by the fundamental Gestalt design principles of alignment, proximity, contrast, and symmetry.

Much of the 2D and 3D design throughout this project was concerned heavily with contrast. The white and black palette was illustrative of form and void; substance and absence. This palette and its underlying theme were used in various fashions as a means of direction and focus. The meditative nature of the VR experience, in particular, called for a silencing of outside distractions. Encircling the exhibit in total darkness helped to transform it into a place of introspection.

The Microscopia logo, like many other graphical aspects of this project, was created in Illustrator. It contains an isometric image of a cube, which commonly represents three-dimensionality. The two upper lines extending out from the center of the cube breach the boundary of the hexagonal shape and form antennae, indicating a relation to insects. Perhaps this logo is not as fitting as it was at the time of its inception, as a tardigrade is not an insect, but technically other small organisms like a tardigrade have antennae-like structures.

The finished logo set the precedent for the design aesthetic of thin, straight, white lines. It influenced, as well, the choice of Barlow Semi-Condensed Light as the primary typeface and its Thin Italic variation as the secondary typeface. A grotesk font family, Barlow Semi-Condensed has a modern and uniform appeal that immediately struck me as scientific and informational.

While most vector graphics in this project were drawn manually, the wireframe-style graphics representing the specimens were rendered from the 3D models themselves in Blender. First, the models were decimated to less than one percent of their polygon count using the Decimation modifier. Next, the Wireframe modifier was added to achieve the desired wireframe effect in 3D. To render this to a 2D vector graphic, Freestyle SVG Exporter—a standard Blender plugin—was used. The resulting SVG was opened in Illustrator and adjusted for color and stroke thickness.

CONCLUSION

This Honors thesis project not only proved can models of entire, near-microscopic organisms be created with SEM photogrammetry, but also provided educational contexts in which these models can be displayed and interacted with. It required research into several contrasting disciplines in order to combine them in a way that likely has not been done before.

SEM photogrammetry, in the workflow described, is a relatively inexpensive way to create 3D models of organisms with microscopic detail. Although the models created in this project were satisfactory, more highly refined models could be created from more images captured. This, of course, would require more time and funding. Additionally, Reality Capture proved to be a powerful, fast, and user-friendly photogrammetry software. Its ability to detect features in SEM images and further align them through the use of control points was vital to the fidelity of the resulting models.

Augmented and virtual reality are emerging as educational tools in both museums and classrooms. I expect that they will only increase in prevalence as the technologies become cheaper and developer communities behind them grow. The capacity for both of these technologies to visually portray information is unparalleled, especially when the information can be interacted with and actively discovered by the user. If I were to develop Microscopia further, I would explore different avenues of additionally interactivity in both AR and VR.

Whether or not I am the one to expand this project further, there is certainly a space and a purpose to develop more experiences incorporating SEM photogrammetry, augmented reality, and virtual reality. Above all else, the culmination of technologies promises to give users a truly unique and perspective-altering educational experience.

REFERENCES

Arevalo, H. Alejandro, and Eleanor Groden. “European Fire Ant - Myrmica Rubra Linnaeus.” *University of Florida - Entomology and Nematology*, Feb. 2019, entnemdept.ufl.edu/creatures/urban/ants/Myrmica_ruba.htm.

Ball, A.d., et al. “SEM-Microphotogrammetry, a New Take on an Old Method for Generating High-Resolution 3D Models from SEM Images.” *Journal of Microscopy*, vol. 267, no. 2, 2017, pp. 214–226., doi:10.1111/jmi.12560.

Hoover, Gregory A. “Black Vine Weevil (Department of Entomology).” *Department of Entomology (Penn State University)*, Nov. 2010, ento.psu.edu/extension/factsheets/black-vine-weevil.

“Introduction to Ground Plane in Unity.” *Vuforia Developer Library*, Vuforia, 2018, library.vuforia.com/articles/Solution/ground-plane-guide.html.

Kuzmin, Vlad. “Full Photogrammetry Guide for 3D Artists.” *80 LEVEL*, 80 LEVEL, 5 Mar. 2018, 80.lv/articles/full-photogrammetry-guide-for-3d-artists/?fbclid=IwAR3oUaVgON60-E71wME8TuYWi74lxoptjwsKSuQZqzsV-EeAqKTMzA9sApE.

Lachambre, Sébastien, et al. “Unity Photogrammetry Workflow.” *Unity*, Unity, 23 June 2017, unity3d.com/files/solutions/photogrammetry/Unity-Photogrammetry-Workflow_2017-07_v2.pdf.

Miller, William Randolph. “Tardigrades.” *American Scientist*, 2 Feb. 2018, www.americanscientist.org/article/tardigrades.

Van de Kerckhove, Eric. “HTC Vive Tutorial for Unity.” *Raywenderlich*, 4 Jan. 2019, www.raywenderlich.com/9189-htc-vive-tutorial-for-unity.

APPENDIX A

Images and Graphics

FIGURE 1: The Microscopia logo.

FIGURE 2: The Microscopia app icon.

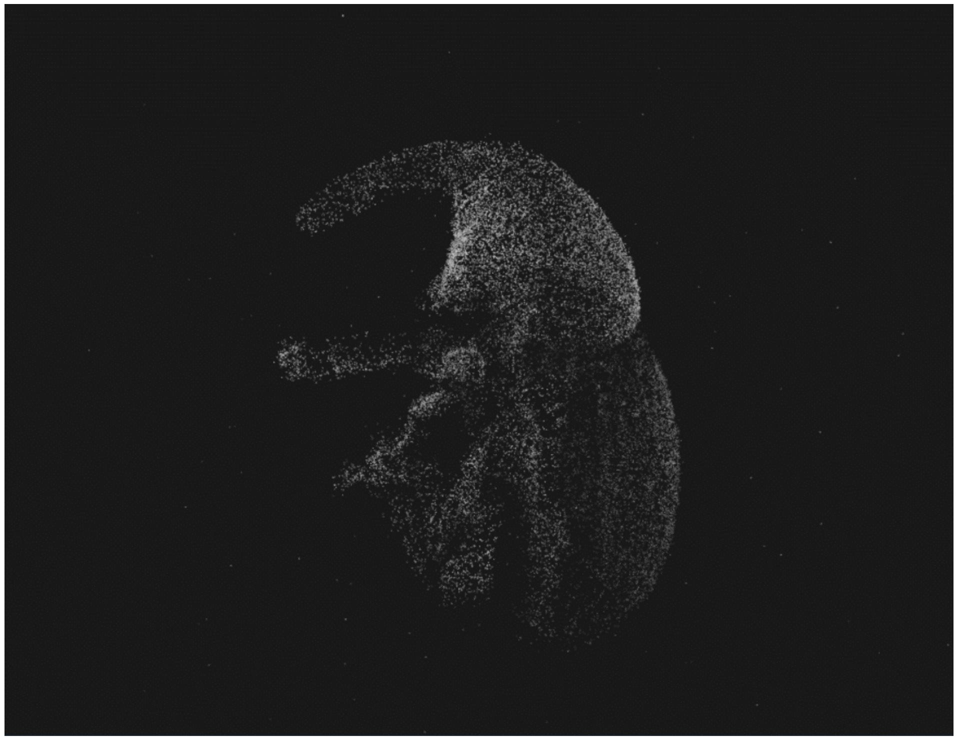

FIGURE 3: The generated 3D point cloud of the black vine weevil specimen.

FIGURE 4: The black vine weevil mesh reconstructed in High Detail and textured.

FIGURE 5: A ventral view of the finalized *Echiniscus virginicus* rendered in Blender.

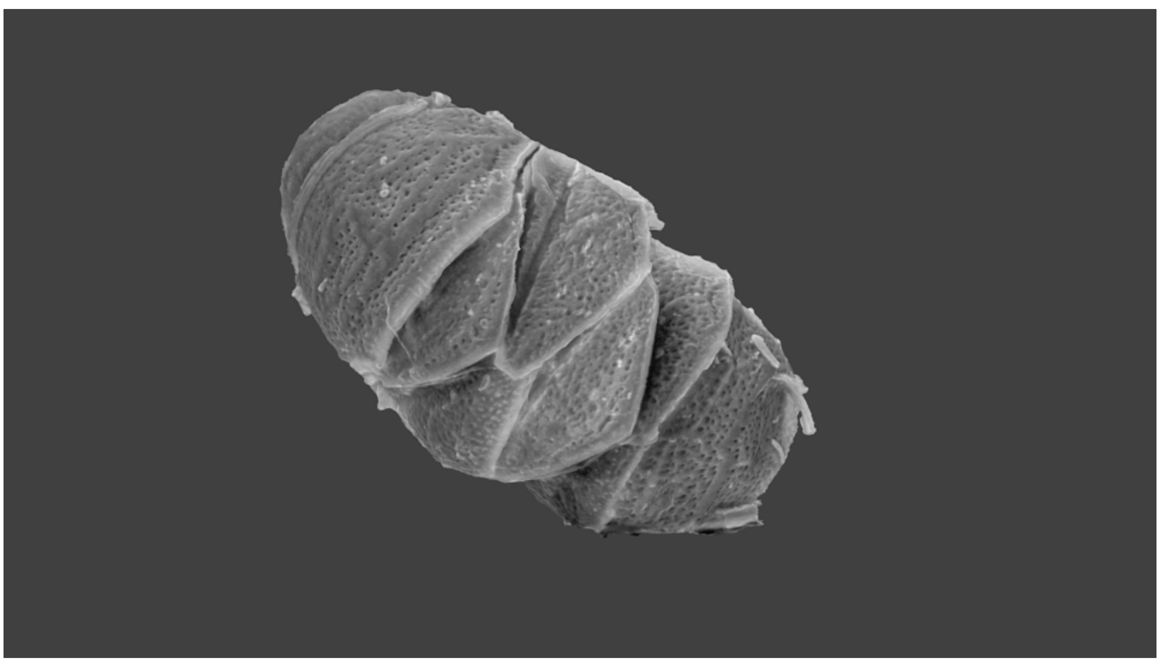

FIGURE 6: A dorsal view of the finalized *Echiniscus virginicus* rendered in Blender.

FIGURE 7: A screenshot of the virtual AP/PE pace within the VR experience.

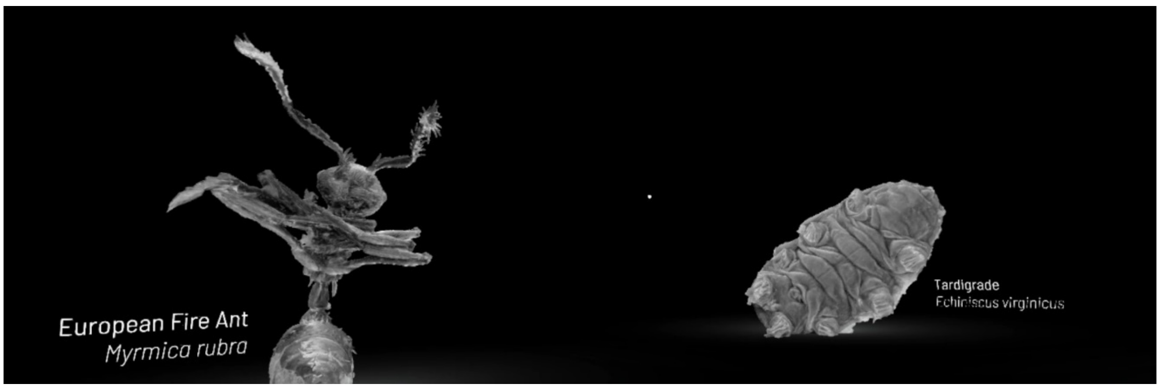

FIGURE 8: A screenshot of the microscopic exhibit within the VR experience.

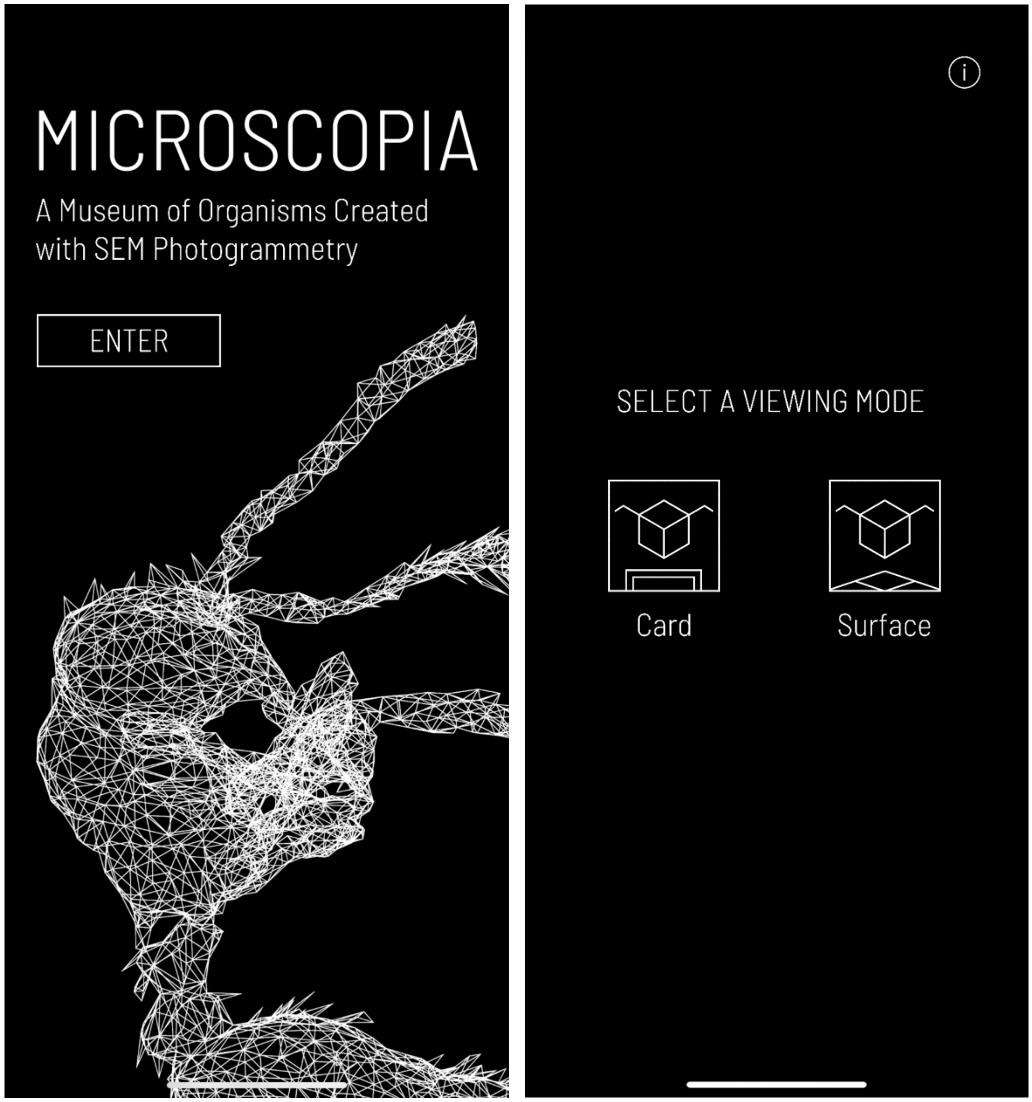

FIGURE 9: Screenshots taken with an iPhone 8 of the title screen and viewing mode menu within the Microscopia app.

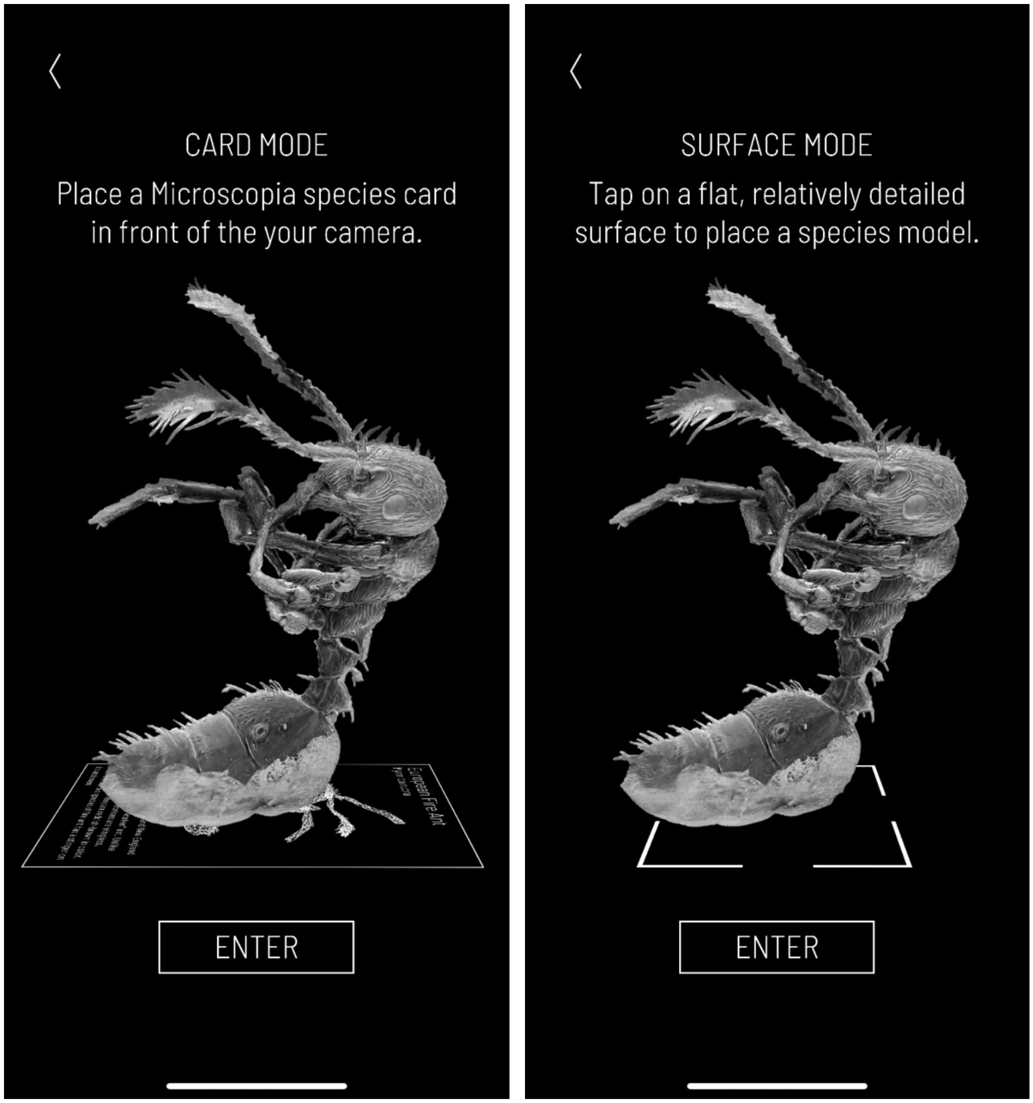

FIGURE 10: Screenshots of both Card Mode and Surface Mode instructional loading screens.

FIGURE 11: A screenshot of Surface Mode with the black vine weevil selected.

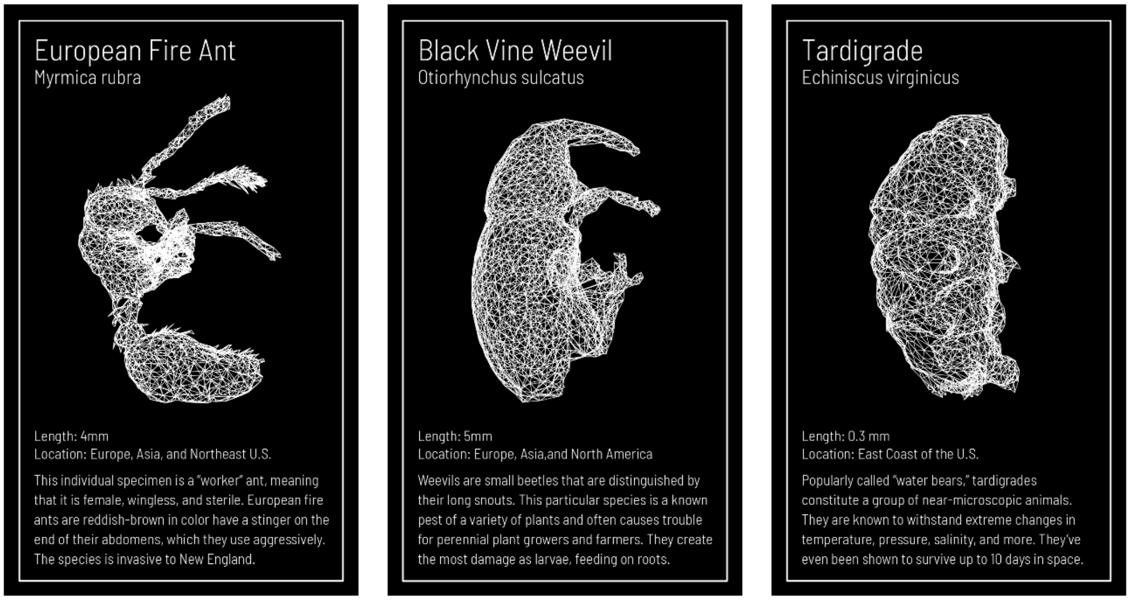

FIGURE 12: Card designs for use with the Microscopia app's Card Mode. They were printed on matte cardstock with dimensions of three inches by five inches. Each has a Microscopia logo on the back.

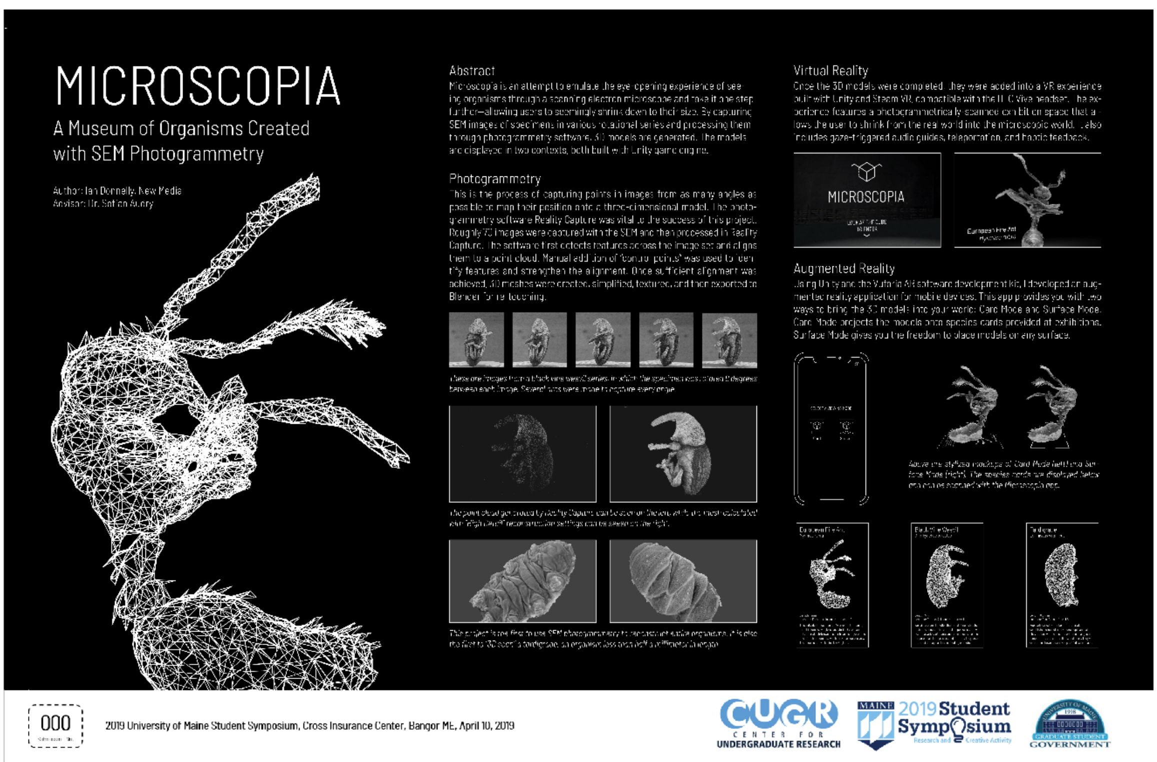

FIGURE 13: The poster presented at the Center for Undergraduate Research's 2019 Student Symposium.

APPENDIX B

Sample C# Scripts

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.EventSystems;
using UnityEngine.SceneManagement;
using Valve.VR.InteractionSystem;

public class CubeRingFill : MonoBehaviour
{
    public Image RingFill;
    public bool shrinkMe = false;
    public bool shrinkStop = false;
    public bool soundIncrease = true;
    public GameObject Player;
    public GameObject mSoundtrack1;
    public GameObject mSoundtrack2;
    GameObject fadeRing;
    public AudioSource noise;
    int lookingAtRing;
    GameObject[] obj;
    public Image shrinkBlack;
    bool loadMicroscopia = true;
    public Material black;

    void Start()
    {
        // Add all elements of the microscopic exhibit to a gameobject
array
        obj = GameObject.FindGameObjectsWithTag("L2");
        // Disable all elements gameobject array on start
        Invoke("BarDeactivate", 0);
        RingFill = GetComponent<Image>();
        GetComponent<Animator>().enabled = false;
        // Disable the fade out until later, when shrinking begins
        shrinkBlack.GetComponent<Animator>().enabled = false;
        // Set the default player scale
        Player.transform.localScale = new Vector3(1.5f, 1.5f, 1.5f);

    }
    void BarDeactivate()
    {
        foreach (GameObject objs in obj)
```

```
        {
            // Disable all elements gameobject array on start
            objs.SetActive(false);
        }
    }

    void BarReactivate()
    {
        foreach (GameObject objs in obj)
        {
            // Activate all elements gameobject array
            objs.SetActive(true);
        }
    }
    // Update is called once per frame
    void Update()
    {
        // Access the camera for variable that says whether or not this
ring is being looked at
        lookingAtRing =
GameObject.Find("VRCamera").GetComponent<LineOfSight>().lookingAtRing;

        if (shrinkMe == false)
        {
            // Fill up the ring when it's being looked at
            // lookingAtRing value determines which ring
            if ((lookingAtRing == 10) && (RingFill.fillAmount < 1))
            {
                RingFill.fillAmount += Time.deltaTime / 1;
            } // Deplete the ring when it's not being looked at
            else if (RingFill.fillAmount >= 0)
            {
                RingFill.fillAmount -= Time.deltaTime / 5;
            }
            // If the ring fills all the way up, the shrinking is
triggered
            if (RingFill.fillAmount >= 1)
            {
                shrinkMe = true;
            }
        }

        if (shrinkMe == true)
        {
            // Increase second track volume when shrinking starts
```

```
            mSoundtrack2.GetComponent<AudioSource>().enabled = true;
            if ((mSoundtrack2.GetComponent<AudioSource>().volume <= 0.75)
&& (shrinkStop == false)){
                mSoundtrack2.GetComponent<AudioSource>().volume +=
Time.deltaTime / 3;
            }

            // After shrink sequence stops, slowly decrease soundtrack
volume
            if ((mSoundtrack2.GetComponent<AudioSource>().volume >= 0.30)
&& (shrinkStop == true)){
                mSoundtrack2.GetComponent<AudioSource>().volume -=
Time.deltaTime / 8;
            }
            // Fade out first track when shrinking begins
            mSoundtrack1.GetComponent<AudioSource>().volume -=
Time.deltaTime / 3;
            RingFill.color = new Color(0f, 255f, 255f, 255f / 255f);
            // Fade out the selection ring
            var tempColor = RingFill.color;
            tempColor.a = 1f;
            tempColor.a -= Time.deltaTime;
            GetComponent<Animator>().enabled = true;
            // Begin fade to black animation on UI Canvas
            shrinkBlack.GetComponent<Animator>().enabled = true;
            if (shrinkStop == false)
            {
                // Gradually decrease the scale of the player
                Vector3 originalScale = Player.transform.localScale;
                Vector3 destinationScale = new Vector3(0.1f, 0.1f, 0.1f);
                Player.transform.localScale = Vector3.Lerp(originalScale,
destinationScale, Time.deltaTime / 4);
            }

            if ((noise.volume <= .5f) && (soundIncrease == true))
            {
                // Increase the volume of the white noise
                noise.volume += Time.deltaTime / 25;
                noise.pitch += Time.deltaTime / 10;
            }
            if ((noise.volume >= .5f) && (soundIncrease == true))
            {
                // When the white noise volume reaches a certain point,
stop increasing
                soundIncrease = false;
```

```
            }
            if ((noise.volume >= 0f) && (soundIncrease == false))
            {
                // When the white noise volume stops increasing...
                // Stop shrinking
                shrinkStop = true;
                // Set the skybox to black material
                RenderSettings.skybox = black;
                // Turn the ambient lighting off
                RenderSettings.ambientIntensity = 0;
                // Set the player back to normal scale
                Player.transform.localScale = new Vector3(1.5f, 1.5f, 
1.5f);
                // Decrease white noise volume to 0 in 1 second
                noise.volume -= Time.deltaTime;
                noise.pitch += Time.deltaTime / 1;
                // Add all elements from the AP/PE space to an array
                GameObject[] gameObjectArray = 
GameObject.FindGameObjectsWithTag("L1");
                // Disable those elements
                foreach (GameObject go in gameObjectArray)
                {
                    go.SetActive(false);
                }
                // Enable microscopic exhibit elements
                Invoke("BarReactivate", 0);
            }
        }
    }
}
```

Model Selection in AR Surface Mode

```
using UnityEngine;
using System.Collections;
using UnityEngine.SceneManagement;

public class SurfaceGUI : MonoBehaviour
{
    public Texture btnTexture;
    public GUIStyle MicroscopiaButton;
    public GUIStyle AntButton;
    public GUIStyle Type;
    public GameObject ant;
    public GameObject weevil;
    public GameObject tardigrade;
    public GameObject groundPlane;
    public int model = 1;

    void Start()
    {
        // Set Ant as selected model at start
        ant.SetActive(true);
        // Set Tardigrade as deactivated at start
        tardigrade.SetActive(false);
        // Set Weevil as deactivated at start
        weevil.SetActive(false);
        model = 1;
    }

    void OnGUI()
    {
        // Create scalable back arrow button
        if (GUI.Button(new Rect(Screen.height/55.4f,
Screen.height/30.9333f, Screen.height/11.11667f/1.5f,
Screen.height/11.11667f/1.5f), btnTexture, MicroscopiaButton))
        {
            // Return to menu if pressed
            SceneManager.LoadScene("Menu Only");
        }
        // Global model variable is set by other button scripts
        if (model == 1){
            // If ant button pressed, set activate ant and deactivate
other models
```

```
            ant.SetActive(true);
            tardigrade.SetActive(false);
            weevil.SetActive(false);
        }
        if (model == 2){
            // If weevil button pressed, set activate weevil and
deactivate other models
            ant.SetActive(false);
            tardigrade.SetActive(false);
            weevil.SetActive(true);
        }
        if (model == 3){
            // If tardigrade button pressed, set activate tardigrade and
deactivate other models
            ant.SetActive(false);
            weevil.SetActive(false);
            tardigrade.SetActive(true);
        }
    }
}
```

APPENDIX C

Narration Script

Ant

Antennae: Ants have poor eyesight and can only detect changes in light, so they primarily use their antennae to navigate their surroundings. Antennae can be also be used to sense food and communicate with other ants.

Mouth: Fire ants have mouthparts designed for chewing and sipping. They don't ingest solid food particles; instead, they grind up food until they can sip it as a liquid. Their mouths even contain strainer-like components called sieve plates that prevent ingestion of solid food.

Stinger: To defend their nests from invaders, Fire ants use the stingers on the ends of their abdomens as a means of attack. The venom from the stinger targets the nervous system, causing an itching or burning sensation. Fire ant stings are often mistaken as "bites" because they use their mouths to latch on to their target while stinging.

Weevil

Snout: Weevils are distinguished by their snouts, called proboscises, that protrude from the front of their heads. The snout contains the mouthparts of the weevil and also has short antennae that are attached on either side.

Legs: As an insect, weevils have six legs. You can see here, however, that this weevil is missing its front legs (called fore legs). This is a product of handling the fragile, dried specimen while posing it.

Reproduction: The black vine weevil is an all-female species. It reproduces through a process called parthenogenesis, meaning that it doesn't need its eggs to be fertilized by a male in order to create offspring. This process produces all-female offspring, and because of this, no male black vine weevils have been found.

Tardigrade

Mouth: Tardigrades have tubular mouths with hard, sharp structures called stylets. These work like teeth to piece plant cells, algae, or small invertebrate. Stylets are lost when a tardigrade molts, but a new set is secreted from a pair of glands on either side of the mouth.

Legs: Each pair of legs on a tardigrade corresponds to a different body segment. The front three pairs are used for motion, while the last pair is often used for grasping onto surfaces. To maintain traction, this species has 4 claws on each of its feet, though other species can have up to 8.

Back Plates: As you can see on the back of this specimen, this species is distinguished by its armadillo-like dorsal plates. These protective structures are made of chitin, the same fibrous substance that is found in the shells of crabs, lobsters, and other arthropods.

AUTHOR'S BIOGRAPHY

Ian J. Donnelly was born in Portland, Maine on October 9th, 1996. He was raised in Windham, Maine and graduated from Windham High School in 2015. Ian is graduating in the spring of 2019 with a degree in New Media and minors in Graphic Design and Biology. He has received a Center for Undergraduate Research fellowship, a New Media grant award, and a College of Liberal Arts and Sciences Senior Recognition award.

Outside of academics, Ian enjoys kayaking, fishing, Nordic skiing, reading, and hiking—particularly in Acadia National Park. After graduation, he plans to pursue a career in user experience and interaction design. He aspires to professionally create interactive exhibits for museums and explore new means of scientific communication.